www.ioactive.com

IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES

Black Ops 2008: It's The End Of The Cache As We Know It

Or: "64K Should Be Good Enough For Anyone"

Dan Kaminsky
Director of Penetration Testing
IOActive, Inc.

Introduction

- Hi! I'm Dan Kaminsky
  - This is my 9th talk here at Black Hat
  - I look for interesting design elements – new ways to manipulate old systems,
    old ways to manipulate new systems
  - Career thus far spent in Fortune 500
    - Consulting now
  - I found a really bad bug a while ago.
    - You might have heard about it.
    - There was a rather coordinated patching effort.
    - I went out on a very shaky limb, to try to keep the details quiet
      - Asked people not to publicly speculate
        - » Totally unreasonable request
        - » Had to try.
      - Said they'd be congratulated here

Thanks to the community

- First finder: Pieter de Boer
  - 51 hours later
- Best Paper
  - Bernard Mueller, sec-consult.com
  - Five days later, but had full info/repro
- Interesting thinking (got close, kept off lists)
  - Andre Ludwig
  - Nicholas Weaver
  - “Max”/@skst (got really *really* close)
  - Zeev Rabinovich
  - Michael Gersten
  - Mike Christian
- Left the lists
  - Paul Schmehl
  - Troy XYZ
  - Others ☺
- Thanks
  - Jen Grannick (she contacted *me)*
  - DNSStuff (they taught me LDNS, and reimplemented my code better)
  - Everyone else (people know who they are, and know I owe them a beer).

Obviously thanks to the Summit Members

- Paul Vixie
- David Dagon
  - Georgia Tech – thanks for the net/compute nodes
- Florian Weimer
- Wouter Wijngaards
- Andreas Gustaffon
- Microsoft
- Nominum
- OpenDNS
- ISC
- Neustar
- CERT

- People have really been incredible with this.
- What did we accomplish?

IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES

There are numbers and are there are numbers

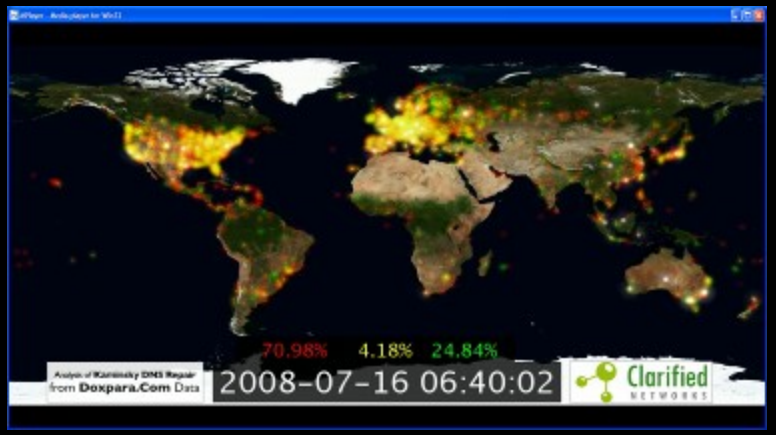
- **120,000,000**
  - The number of users protected by Nominum's carrier patching operation
  - They're not the Internet's most popular server!
    - That's BIND, and we saw LOTS of BIND patching
  - They're not the only server that got lots of updates
    - Microsoft's Automatic Updates swept through lots and *lots* of users
    - Do not underestimate MSDNS behind the firewall. Just because you can't scan for it doesn't mean it's not out there.
  - One hundred and twenty million – that, alone, is 42% of broadband subscribers. That's cool.
- That was a lot of work. The IT guys stepped up.
  - Lets watch 'em!

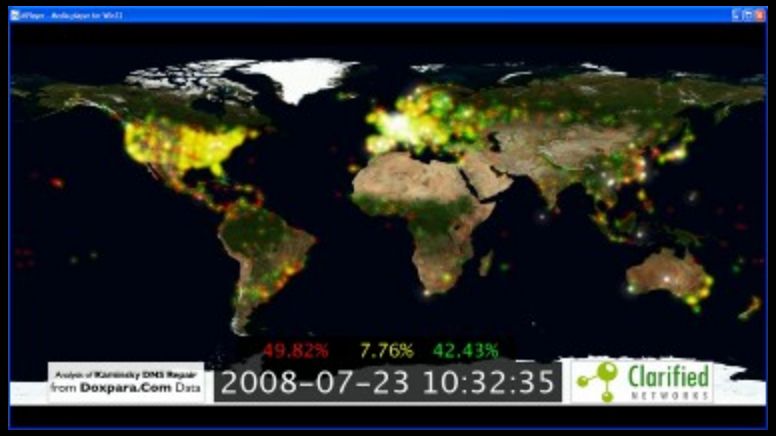
What about the Fortune 500?

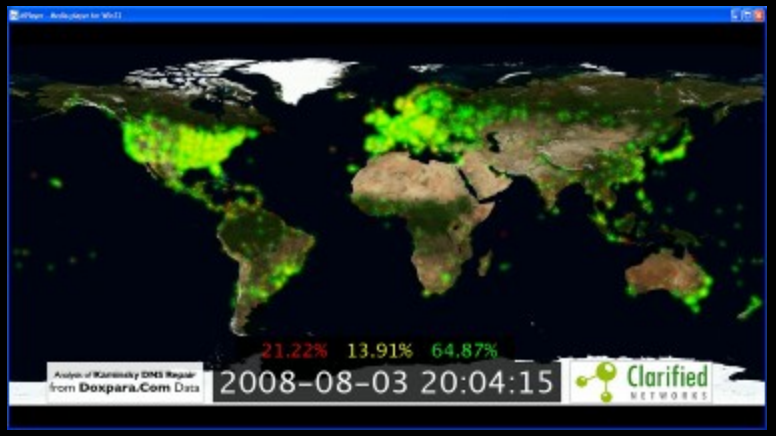
- Doing OK!
  - From significant sample of Fortune 500:
    - 70% have tested and patched mail servers
    - 15% patched, but suffer from NATs
      - This has been a real problem
    - 15% unpatched
  - Non-mail servers doing almost as well
    - 61% patched
    - 21.75% patched, but suffer from NATs
    - 17.25% unpatched
- Thank you, Mike Ryan, Jacob Carlson, Charles Henderson of Trustwave R&D, Application Testing
  - They do PCI testing, and were well positioned to encourage testing and validation for this flaw
  - Certainly an unusual and awesome use of security through compliance
    - Unprecedented – probably a better hack than the original bug

IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES

Can we watch the patching in action? (Thank you, Joachim Viide et al, Clarified Networks)







But why all this work?

- Good question

Intro to DNS

- System on Internet which maps names (that humans understand) to numbers/ “IP Addresses” (that the Internet can deal with)
  - Just like 411 information, or the White Pages
  - Numbers change so frequently on the Net, that it’s easier to just keep looking them up
  - Almost everything on the Internet depends on DNS returning the right number for the right request
    - More than you’d think
      - Foreshadowing!

DNS is distributed

- Three possible answers to any question
  - "Here's your answer"
  - "Go away"
  - "I don't know, ask that guy over there"
    - This is delegation. You start with a request, and then get bounced around all over the place.
    - 13 root servers: "www.foo.com? I don't know, go ask the com server, it's at 1.2.3.4"
    - Com server: "www.foo.com? I don't know, go ask the foo.com server, it's at 2.3.4.5"
    - Foo.com server: "www.foo.com? Yeah, that's at 3.4.5.6."
- Dealing with "ask that guy" ("Delegation") a lot of work, so DNS infrastructure divided into Servers (that run around) and Clients, or "Stub Resolvers", that either do or don't get an answer
  - BIND = Name Server
  - Your Desktop = Stub Resolver

What about bad guys?

- If everything depends on receiving the right number for the right name, wouldn't a bad guy want his number returned instead?
  - Yup
- So when the name server asks ns1.foo.com for www.foo.com, couldn't the bad guy reply first, with his own number?
  - Yup
- What's supposed to prevent this?
  - Transaction ID – "random" number between 0 and 65535. The real name server knows the number, because it was contained in the request. The bad guy doesn't know – at best, he can guess

The Guessing Game

- Good guy – the real name server – has a 65,536 to 1 advantage over the bad guy
  - Those are long odds for the bad guy
- When the good guy gets his reply in – “wins the race” – he can say how long until the next “race”, via something called the TTL, or “Time To Live”
  - 1 minute
  - 1 hour
  - 1 day
  - This is how long a given number is “valid” for a particular name.
- 1 day * 65,536 races / 2 = 84.5 years for 50% chance
  - Good luck on that.

And thus, Forgery Resilience

- Document being assembled by Bert Hubert, author of PowerDNS
  - Was soon to be an Internet RFC
- Basic concept: Long TTL = High Security, Low TTL = Low Security
  - 65,535 minutes / 2 = 22 days for 50% chance
- The basic concept is wrong, *very very wrong*
  - Quote from my Black Hat 2007 talk: "TTL's are not a security feature"
  - The concept implies its opposite, i.e. that the bug I found must exist, because there's no way something not intended to be a security feature would ever stand up to attack
    - So Bert delayed his RFC while we fixed the bug
- However, I had no idea this was under development when I found the flaw
  - So what's the bug?
    - There are three issues – first two were kind of known, the last is what's new

First: If it's a race, between who can reply with the correct TXID first, the bad guy has the starter pistol

- Bad guy can force the name server to go run to the good guy and look something up
  - It takes time to get the real request (with random number) to the good guy
  - It takes more time to get the real response back from the good guy
  - It takes no time for the bad guy to immediately follow up a request with a fake response
    - Might have the wrong random number, but it'll definitely arrive first

Second, who said the bad guy can only reply once

- Winner of the race is the first person to show up with the correct random number
- Nowhere does it say the bad guy can't try lots of random numbers
  - He has time – he doesn't need to wait for anything to reach him, because nothing ever will
- If the bad guy can reply 100 times before the good guy returns, that 65536 to 1 advantage drops to 655 to 1.
  - Alas…still long odds. And when he loses, he has to wait the TTL. That could be 655 days – almost 2 years!
  - Or maybe not.

Finally, the bad guy doesn't actually need to wait to try again.

- If the bad guy asks the name server to look up www.foo.com ten times, there will only be one race with the good guy
  - The first race will be lost (most likely), and then the other nine will be suppressed by the TTL
    - No new races on this name for one more day! Here, use the answer from a while ago
    - So, can we race on other names?
- If the bad guy asks the name server to look up 1.foo.com, 2.foo.com, 3.foo.com, and so on, for ten names, there will be 10 races with the good guy
  - TTL only stops repeated races for the same name!
- Eventually, the bad guy will guess the right TXID before the good guy shows up with it
  - And now…the bad guy is the proud spoofer of … 83.foo.com
  - So? He didn't *want* to poison 83.foo.com. He wanted www.foo.com

Bait and Switch

- Is it possible for a bad guy, who has won the race for 83.foo.com, to end up stealing www.foo.com as well?
  - He has three possible replies that can be associated with correctly guessed TXID
  - 1) “Here’s your answer for 83.foo.com – it’s 6.6.6.6”
  - 2) “I don’t know the answer for 83.foo.com.”
  - 3) “83.foo.com? I don’t know, go ask the www.foo.com server, it’s at 6.6.6.6”
    - This has to work – it’s just another delegation
      - 13 root servers: “83.foo.com? I don’t know, go ask the com server, it’s at 1.2.3.4”
      - Com server: “83.foo.com? I don’t know, go ask the foo.com server, it’s at 2.3.4.5”
      - Foo.com server: “83.foo.com? I don’t know, go ask the www.foo.com server, it’s at 6.6.6.6”

Enter The DNSRake

- Named after a common method for lockpicking
- 1) Send a query to a nameserver, for $RANDOM.foo.com
  - The bad guy has the starter pistol
- 2) Send 200 fake replies to that nameserver, with TXID 0-200
  - The bad guy can reply multiple times
- 3) Send replies containing nameserver redirections to www.foo.com
  - $RANDOMwww.foo.com IN NS www.foo.com
    www.foo.com IN A 6.6.6.6
  - If this works, it works
  - If it fails, return to step 1

What's it look like?

- 1 0.000000 1.2.3.4-> 66.240.226.139 DNS Standard query ANY 2465786792ask-dan-at-foo-com.foo.com
- 2 0.000669 1.2.3.4-> 66.240.226.139 DNS Standard query response NS ask-dan-at-foo-com.foo.com A 6.6.6.6
- 3 0.001008 1.2.3.4-> 66.240.226.139 DNS Standard query response NS ask-dan-at-foo-com.foo.com A 6.6.6.6
- 4 0.001304 1.2.3.4-> 66.240.226.139 DNS Standard query response NS ask-dan-at-foo-com.foo.com A 6.6.6.6
  - 5-201 are like 4
  - 202 repeats back to 1

Running the attack...

- dnsrake 66.240.226.139 1.2.3.4 ask-dan-at-foo-com.foo.com 63752 6.6.6.6 200
  - 1) IP of my name server, mail.doxpara.com (BIND9, but it'll work against anyone)
  - 2) IP of ns*.foo.com
    - Repeat command for each ns*
  - 3) Name I'd like to pollute
  - 4) Fixed source port of my server, leaked by having it look up something off one of my own domains
  - 5) IP I want to force people to use
  - 6) Ratio of random requests to spoofed responses

Validating the attack

- # dig @mail.doxpara.com ask-dan-at-foo-com.foo.com
  ; <<>> DiG 9.2.5 <<>> @mail.doxpara.com ask-dan-at-foo-com.foo.com
  ; (1 server found)
  ;; global options: printcmd
  ;; Got answer:
  ;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 59212;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 5, ADDITIONAL: 5
  ;; QUESTION SECTION:
  ;ask-dan-at-foo-com.foo.com. IN A
  ;; ANSWER SECTION:
  ask-dan-at-foo-com.foo.com. 86279 IN A 6.6.6.6

Extending The Attacks

- So that works against pretty much everything in wide deployment
  - BIND8/9
  - MSDNS
  - Nominum (with some tweaks)
  - Doesn't work against DJBDNS, PowerDNS, MaraDNS
- Most commonly offered defense: "Our DNS servers don't accept queries from the outside world. They must be safe!"
  - Can someone ask them to look up www.doxpara.com, will they return 157.22.245.20?
  - If so, don't be so sure

On Bailiwicks

- 1.foo.com is able to return a reply for www.foo.com for a reason
  - "In bailiwick"
  - The root servers can return any record
  - The com servers can return any record…for com
  - The foo.com servers can return any record for foo.com
  - It wasn't always this way, but then Eugene Kashpureff wanted his own TLD (com, net, etc)
    - He just added additional records in every reply for foo.com, declaring his own TLD existed
    - Everyone accepted it
  - So the bailiwick system was invented to prevent foo.com from declaring anything about com, or some other new TLD
  - (This was 1997, the last time we had a bug this bad.)
    - 2002, Vagner Sacramento's Birthday Attacks, couldn't override cache
    - 2007, Amit Klein's TXID prediction, couldn't override cache

Out Of Bailiwick Referrals, or How To Attack Name Servers Behind Firewalls

- DNS doesn't stop working when you get a referral into another bailiwick
  - If foo.com says "Ask that guy over there, here's his address", and that guy is bar.com, the name server goes back to the root and asks: "Heh, where's bar.com?"
- This means any lookup can spawn any other arbitrary lookup, on demand
  - 1. Force a lookup to 1.badguy.com
  - 2. Reply with a referral (NS or CNAME) to 1.foo.com
    - This *immediately* causes a request to be sent to the foo.com name server
  - 3. Follow the reply with an immediate stream of fake replies from the foo.com name server
- There are many many ways to do #1

The Many Starter Pistols Of Mr. Bad Guy

- Web Browsers will look up what the bad guy wants
  - Any link, any image, any ad, anything can cause a DNS lookup
  - No Javascript required, though h0h0h0 it helps
- Mail Servers will look up what the bad guy wants
  - On first greeting: HELO
  - On first learning who they're talking to: MAIL FROM
  - On SPAM check
    - Get worried now.
  - When trying to deliver a bounce
  - When trying to deliver a newsletter (Lyris, ahem, plz patch)
  - When trying to deliver an actual response from an actual employee

GetHostByName() Considered Harmful

- Web log resolution
  - Reverse DNS – given a connection from 6.6.6.6, PTR lookup to www.badguy.com
  - Return a CNAME (alias) with a 0 TTL, to anyone else's name
  - Each record will now repeatedly look up the attacker controlled name, even though the target aliased into has a longer lifespan
- "Web Bugs" in documents
  - File formats that "call home" to their authors upon reading
  - They're not just about privacy violation anymore
- Lots and lots of things in Web 2.0
  - URL attachments
  - Keep getting more worried

GetHostByAddr() ain't doing too well either

- IDS/IPS
  - Scan a large network, something will automatically try to figure out who you are
    - Reverse DNS maps numbers to names, using PTR lookups
      - Start trembling
    - PTR can reply with a CNAME or NS too
    - Technically you don't even need to scan with your own IP address
      - Heh, you know they'll be looking, though maybe you don't know exactly when.

Roy Arends' Trick

- The Microsoft nameserver, when it sent a query to the outside world, would accept queries back on that particular socket
  - So, you answer a question with a question
  - Roy found this, mentioned it to the dev, it was fixed in an unrelated codepath and the fix was absorbed with the overall update
  - Nice find, Roy!

About Those Internal Only Name Servers: An amusing trick

- Badguy wants to trigger a lookup on a nameserver at 100.1.2.3
- Badguy spoofs source IP to be…100.1.1.3
- This passes the router ACL, because it's a nearby subnet
- This passes the "internal-only" recursion rule, because it's a nearby subnet
- Badguy will never see the response to his query, because that's going back to 100.1.1.3
- But Badguy asked for 1.badguy.com.
  - He'll see the request for 1.badguy.com
  - Which he can reply to with…1.badguy.com IN CNAME 1.foo.com, and follow up with a flood of fake replies for: 1.foo.com IN NS www.foo.com, www.foo.com in A 6.6.6.6

The "Fix", As Per DJB: Source Port Randomization

- Before: 65536 to 1 odds
- After: Between 163,840,000 to 1 and 2,147,483,648 to 1 odds
- This is an improvement
  - That's *a lot of traffic to go unnoticed*
    - Not necessarily too much
- So why not go with something perfect?

THERE ARE MANY, MANY VARIANTS OF THIS ATTACK

- As said in my Black Hat 2007 talk: **TTL is not a security feature**
  - There are many, *many* ways around the TTL logic
- Family 1: Alternate Referrers
  - CN AME
  - DNAME
  - Extra Glue
- Family 2: Request Polluters (Cause request to be ignored upstream)
  - Unknown QTYPE
  - Unknown QCLASS
  - Nonexistent Names

Florian Weimer / Brian Dowling's new PowerDNS attack

- In short, PowerDNS does not respond to certain queries it considers malformed. This in itself is not a problem, and was even thought of as a security measure. Brian and Florian, independently I think, have discovered that not answering a query for an invalid DNS record within a valid domain allows for a larger spoofing window of the valid domain. Because of the Kaminsky-discovery, this has become bad. For a sophisticated attacker, this provides no benefit. However, such a long window allows unsophisticated hackers to achieve better results.
  - Bert Hubert, PowerDNS
- Basically, recursive resolvers would pass a query to PowerDNS, which it authoritatively would ignore. This meant that there was an infinite window – the other guy wasn't even in the race

And Keep Going…

- Other methods
  - Anything that causes the cache to clear
  - Abusing naturally low-TTL records
  - Abusing IDS systems that block if they see an attack (2005 Black Ops)
  - Just polluting a subdomain, because web browsers don't like attacker controlled subdomains (2008 Toorcon talk, i.e. Rickrolling The Internet)
  - AND MORE

The Choice

- 1) Point fixes, where we're guaranteed to miss at least something
- 2) A generic, sledgehammer fix, that applies a minimum level of security to vulnerabilities we don't know about yet
  - Yes, I was even being sneaky about what I meant by Sledgehammer fix ☺
- **DJB WAS RIGHT**
  - **NOT PERFECT – he has bugs too, as we're seeing (and patching, don't ask)**
    - For example, he didn't implement birthday attack protection – he believed port randomization was enough
    - There may be other issues
  - Everyone should be better, even DJBDNS – we all need to get to work
  - Thus begins a debate on alternative solutions
    - Lets not have it with a backdrop of 10 second kills every time a gap is identified

The Caveat

- Many solutions will be proffered
- **Unless it is compatible with the Root Servers and the TLD's – either by backwards compatibility or by code modification to those particular authoritative servers, no other modification is helpful**
  - It doesn't matter how you lock down foo.com if .com is broken
- There's a blog entry at http://www.doxpara.com going through all the other solutions, and how they're all vaguely broken
  - I'm here to talk about the *present* bugs

What of the client?

- Hasn't been the focus of the remediation efforts
  - Has received two MSRC fixes in the last six months:
  - 1) TXID fix, for Amit Klein's research
  - 2) Source Port fix, for my research
  - Why?

On Amit's Client TXID Research

- A lot of people wondered why his TXID bug in Client got fixed
  - Given TXID (Random #) TX1, and TX2, predict TX3
  - That's an important thing to be given!
- So people were complaining that they couldn't figure out how to execute Amit's attack in the real world
  - If you can sniff TX1, you don't need to guess TX2, you can also sniff TX2
  - Suppose you have a nameserver that forwards TX1, unmodified, from the client through itself to an upstream server. Either you can see TX1, in which case you can reply to it, or you can't, but you can send your own TX2, and reply to it.
    - "Hello lame forwarder. Please ask for www.foo.com with txid 100. By the way, here's a reply for www.foo.com on txid 100. What a coincidence, that is the correct txid!"
- Amit caught some flak. Too bad he was right ☺

Nothing Can Be Analyzed In Isolation

- To understand how to attack Amit's bug, you must realize that things *happen* before, and after a DNS reply
  - Before: A HTTP request is sent to before.badguy.com
    - Say, on port 10000
    - This retrieves HTML, with an IMG, which asks for after.badguy.com
    - DNS request goes out…on port 10001
      - This was pre-MSRC ☺
  - After: A HTTP request is sent to after.badguy.com
    - When? Immediately
    - Where? Wherever after.badguy.com claims to be
    - **Where and when are signals. They are low bandwidth signals…but we're only trying to collect a 16 bit value!**

The Chain

- Client browses to a website controlled by badguy
- Client looks up after.badguy.com against Local Name Server
- Local Name Server goes to ns1.badguy.com, asking for after.badguy.com
- ns1.badguy.com **doesn't reply** to Local Name Server.
- Local Name Server thus **doesn't reply** to Client.
- Client sits around twiddling its thumbs.
  - Open port
  - Open TXID
- Ns1.badguy.com pretends to be Local Name Server, replying with all possible TXIDs
  - Knows what port from the before.badguy.com connection
  - Doesn't know what TXID – but will, on average, guess every 32K packets
    - If doesn't guess in time, oh well, try again
  - Will eventually guess correctly

Signals

- When ns1.badguy.com guesses the TXID correctly:
  - 1) A connection will go to whatever address ns1.badguy.com declared for after.badguy.com
  - 2) This will happen immediately
- Strategies
  - 1) Timing: *When* the HTTP connection arrives, see what we were sending a few milliseconds ago.
    - 100ms or so of accuracy, on a 3 second window, yields ~5 bits of data
    - Can retry
  - 2) Direction (thank you Florian Weimer): The more addresses the bad guy has, the more information he can get via which address "wins".
    - If he has 1024 addresses, he gets 10 of 16 bits
    - If he has 65536 addresses (a class B), he wins
    - Without IPv6, that's not happening..

Shared Signals

- There are lots of hosts out there
- Can an attacker borrow some of them?
  - Return the IP addresses of other hosts, then find out which one was “visited”?
- Alternate approaches:
  - 1) Idle Scanning
    - Find 64K boxes that don’t have any traffic
    - Forge replies across all TXIDs, each with a different address destination
    - Check all boxes for sudden traffic spikes
    - Doesn’t really work anymore, too many people scanning the Internet ☺

Another Path

- 2) PTR pollution
  - Find 64K networks that reverse lookup everything, and put it into a publicly visible name server
  - Forge replies, then see what's in the name server cache
  - Too noisy, can't be run multiple times in short succession

Nobody ever expects The Billy Hoffman Option

- 3) Return 64K different web servers, then read via the DOM which one you actually hit
  - Whoever it is, is going to be a host in your own domain
  - DNS rebinding to discover DNS TXID
    - Huzzah!
  - Can also find 64K sites with different cookies, and then analyze document.cookie to see which one was hit

Of course, much easier with my attack

- Don't try to guess next TXID, just force repeated lookups for a name and then fill in whatever answer you want
  - * - Initiate sequence to trigger a dns lookup by the adns resolver. Send * the same range of spoofed DNS ids in a constant flood spoofed as the * primary DNS server for the host. Even a local DNS request will take * long enough to allow some amount of the spoofed DNS responses through * before the primary DNS responds. Since the resolver does not cache * results, the dns lookups can be triggered until the DNS id is * incremented within the DNS id range being spoofed.
    - h0dns_spoof.c - zmda - saik0pod@yahoo.com
- "Sniper Rifle" vs. a Nuke
  - Attack one host? Or many?

So, is that all?

- No. That's HOW to attack DNS. More interesting question: WHY to attack DNS.

We Start With The TLDs

- It is indeed possible to pollute com, net, org, etc.
  - Directly: com NS
  - Indirectly: A.GTLD-SERVERS.NET, B.GTLD-SERVERS.NET., C.GTLD-SERVERS.NET…
- When the bad guy poisons com, he gets all requests
  - Even requests he didn't know in advance he wanted!
  - He gets to decide:
    - What he'll poison forever (response, long TTL)
    - What he never wants to see again (delegation, real NS)
    - What he'll check out for a little while (response, short TTL)

MX Intercept: It's Not Just For the NSA Anymore

- "Remember how pissed you were when you found out the NSA had rooms where they could read everything? That's every kid right now." –Brad Hill
- Mail is special – has its own type of record
  - MX – Mail Exchange
- Attacker who owns com, can see who's sending mails to who, and can pick off any he likes
  - Can silently intercept, then let the mail run off to its correct destination
    - Give himself top priority, fail to fully accept a message, then let the message fall through to the next server

Message Pollution

- 1/3rd of attacks come from direct user action
  - Loading a document
  - Downloading and installing malware
- Attacker can also accept a message, infect attachments with malware, and forward it along
  - DOC -> Infected Doc
  - EXE -> Infected Exe
  - ZIP with Password containing EXE -> ZIP with Password containing Infected EXE
    - Attacker can read ☺
  - Link to EXE -> Link to infected EXE
    - Attacker can either change link, or poison link in destination

Shouldn't The SPAM Filter Stop This?

- SPF should notice the wrong IP
  - SPF comes from DNS
  - All SPAM filtering comes from DNS
  - Can actually hijack SPAM filters – attacker ends up controlling mail reception entirely

Not going there, but…

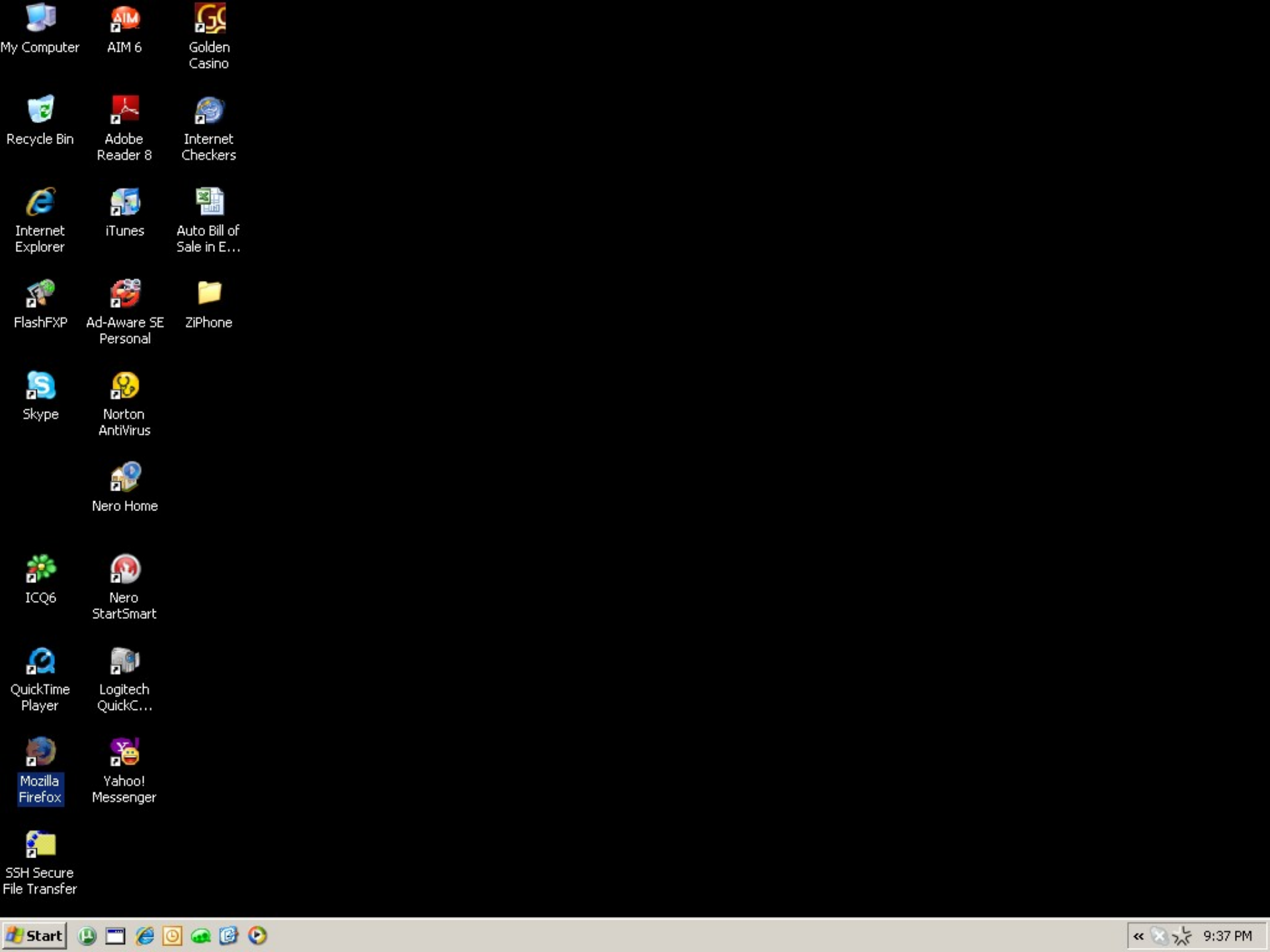
- SIP ain't looking too great either
  - SRV records are easily detectable
  - SIP INVITE/REGISTER messages look like they can contain DNS names – triggering a lookup in target networks
    - If you have an environment that explictly uses DNS name contacts, you might even be able to choose your intercepts
- Thanks to Zane Lackey

Spidey Sense

- Obviously the entire web is affected, for a client behind a corrupted DNS server
  - Can directly poison via com corruption
    - Requires rebinding to read actual site contents
  - Can indirectly poison a single site via its subdomain library dependencies
    - Prototype.js
    - CSS scripts
  - Can indirectly poison the entire web via google-analytics.com, ad.doubleclick.net, sitemeter, or any other codebase commonly loaded via an external <script src> tag.
  - Hope you're not downloading any executables from the web…
- (But SSL will save us!)

The Internet is more than the Web; HTTP is more than the Browser

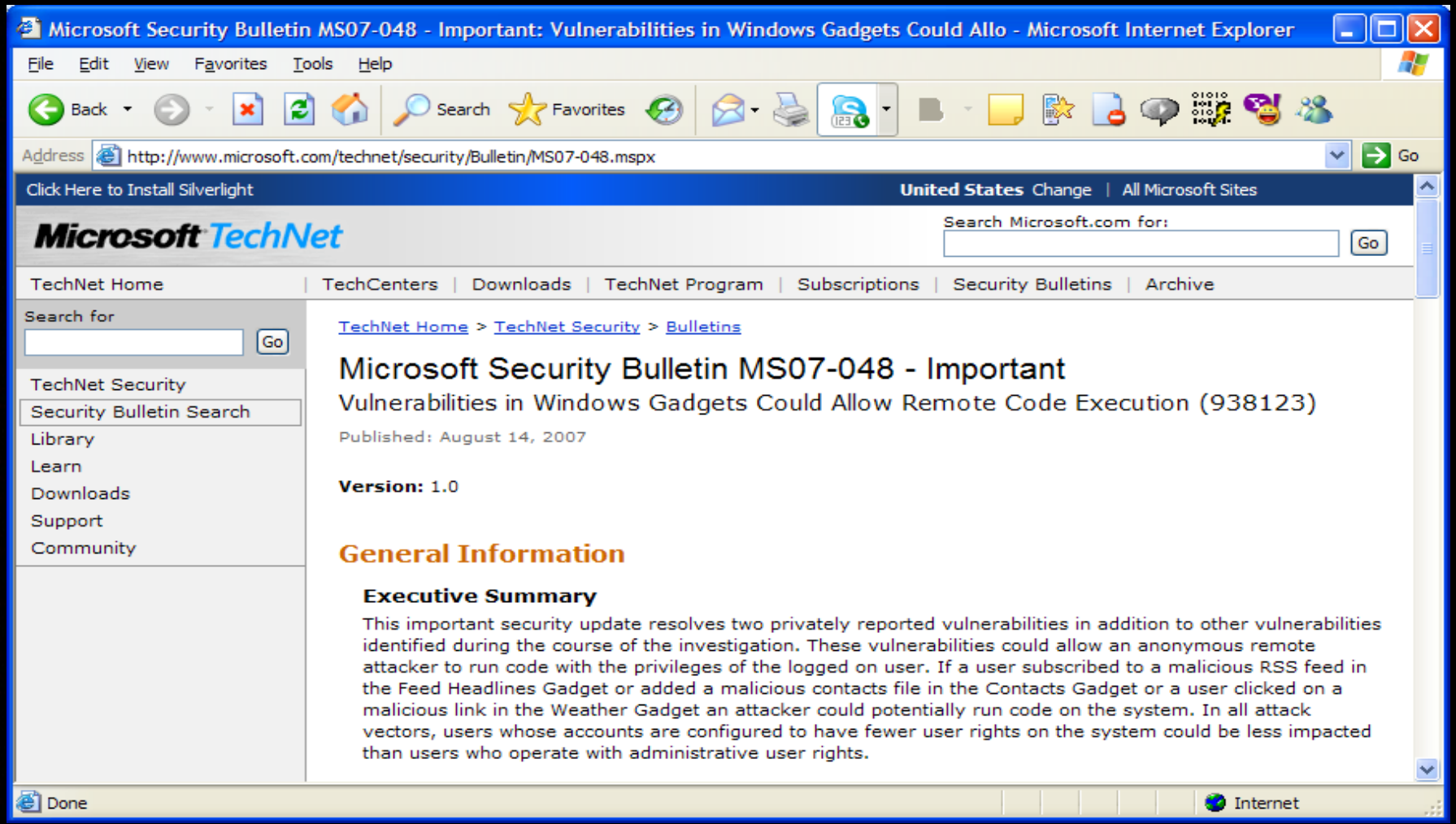
- Welcome to the third age of hacking
  - 1st age: Servers
    - FTP, Telnet, Mail, Web, Time
    - These were the things that consumed bytes from a bad guy
    - These are the things that got locked down
  - 2nd Age: Browsers
    - Javascript, ActiveX, Java, Image Formats, DOMs
    - These are the things that are getting locked down
      - Slowly
      - Incompletely (ActiveX Sitelock to http:// doesn't work too well right now)
  - 3rd Age: EVERYTHING ELSE
    - Check out this desktop from an Internet Cafe


My Computer
AIM 6
Golden Casino
Recycle Bin
Adobe Reader 8
Internet Checkers
Internet Explorer
iTunes
Auto Bill of Sale in E...
FlashFXP
Ad-Aware SE Personal
ZiPhone
Skype
Norton AntiVirus
Nero Home
ICQ6
Nero StartSmart
QuickTime Player
Logitech QuickC...
Mozilla Firefox
Yahoo! Messenger
SSH Secure File Transfer
Start
9:37 PM

We're no longer in browserland anymore…

IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES

Remember Sidebar from Last Year?



This is not an exception

- Browsers are *really really good client code*
  - Relatively ☺
  - They're so much more complex than anything we ever put on the server
  - We've been trying to secure them for far longer
- What do you think happens when you fuzz weak clients?

Ilja van Sprundel, dumb fuzzing IRC with ircfuzz.c

- * ircfuzz v 0.3 by Ilja van Sprundel. * so far this broke: - BitchX (1.1-final) * - mIRC (6.16) * - xchat (2.4.1) * - kvirc (3.2.0) * - ircii (ircii-20040820) * - eggdrop (1.6.17) * - epic-4 (2.2) * - ninja (1.5.9pre12) * - emech (2.8.5.1) * - Virc (2.0 rc5) * - TurboIRC (6) * - leafchat (1.761) * - iRC (0.16) * - conversation (2.14) * - colloquy (2.0 (2D16)) * - snak (5.0.2) * - ircle (3.1.2) * - ircat (2.0.3) * - darkbot (7f3) * - bersirc (2.2.13) * - Scrollz (1.9.5) * - IM2 * - pirch98 * - trillian (3.1) * - microsoft comic chat (2.5) * - icechat (5.50) * - centericq (4.20.0) * - uirc (1.3) * - weechat (0.1.3) * - rhapsody (0.25b) * - kmyirc (0.2.9) * - bnirc (0.2.9) * - bobot++ (2.1.8) * - kwirc (0.1.0) * - nwirc (0.7.8) * - kopete (0.9.2)
- Things are a *little* better now
  - Not much
  - You *really really don't want to be talking to a malicious IRC server*
    - Lets not even talk about netsplits

Lets not forget about the biggest, most extensive clients out there

- Games
  - Gaming - The Next Overlooked Security Hole
    - Ferdinand Schober, Security Researcher
  - We're now seeing those unifying technologies the web, and monolithic engines making their way in to these games. Automatic updates, electronic publishing systems, in-game advertisements, pay-for-item MMORPG systems all of these represent structural weaknesses that more and more people should be exploiting. Given the expectation of today's gamers a far as graphics, physics, and other frivolous crap, smaller developers have to purchase someone else's engine to get started and all of the bugs that come with it.

How do you know what to attack?

- You know where it's going
  - It tells you via DNS
  - That often tells you who it is
- But if you need more, and it's speaking HTTP…
  - Host header
  - User-Agent
  - Extra Headers

Who needs an exploit? Lured by design, upgraded by design

- Francisco Amato's EvilGrade
  - Implemented modules: ---------- - Java plugin - Winzip - Winamp - MacOS - OpenOffices - iTunes - Linkedin Toolbar - DAP [Download Accelerator] - notepad++ - speedbit
  - Bigger companies than I thought. But otherwise, yeah, we knew this was going to be a problem
  - Actually warned LinkedIn in advance

Autoupgrade Is Hard

- To succeed, your update package must be:
  - Signed.
  - Signed by you.
  - Signed by you, using the right EKU (Extended Key Usage)
  - Signed from an unrevoked signature
  - Be the same product
  - Be a new version
- Or you could use SSL, but ZOMG PERFORMANCE
- Translation: Must be Windows Update, or you're hosed ☺
  - Maybe Adobe. MAYBE.
  - See also: "**Secure Software Updates: Disappointments and New Challenges", Bellissimo, Burgess, Fu**

IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES

facepalm

Package Managers As Achilles Heel

Posted by timothy **on Thursday July 10, @06:53PM**
from the **dip-your-computer-in-the-styx** dept.

An anonymous reader writes

> "Researchers from the University of Arizona have released a study that takes a look at the security of ten popular package managers. They were able to show all ten were vulnerable to attacks from a mirror or man-in-the-middle that allow an attacker to (along with other things) crash the system or obtain root access. Furthermore, the researchers created a fictitious administrator and company name and were able to lease a server and get it listed as an official mirror for all the distributions they tried (Ubuntu, Debian, Fedora, CentOS, and OpenSUSE). This raised the question: What keeps you up at night, the thought of attacks on your package manager or previously discussed and patched vulnerability in DNS?"

justin samuel (one of the Arizona researchers) also points out a synopsis on CERT's blog.

- What is this “OR” you speak of?
- What, I can only use one bug at a time?
- DNS **provides** the Man-In-The-Middle that breaks Package Managers!
- But SSL! SSL WILL SAVE US!

Make no mistake

- This is the first big test of SSL
- Has it stood up because of the strength of its crypto?
- Or has it stood up because nobody had the opportunity to play MiTM?

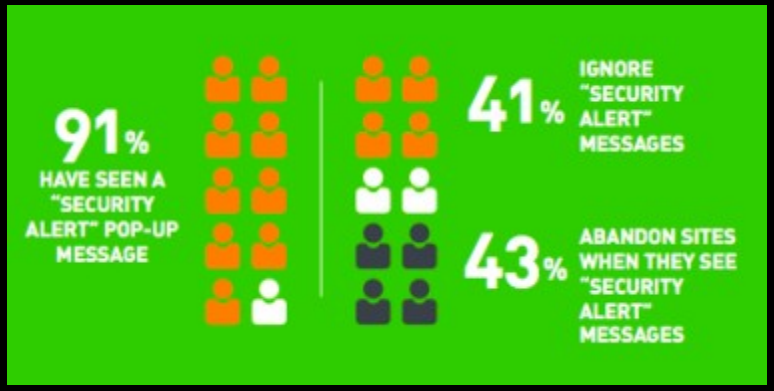
Lets talk about SSL.

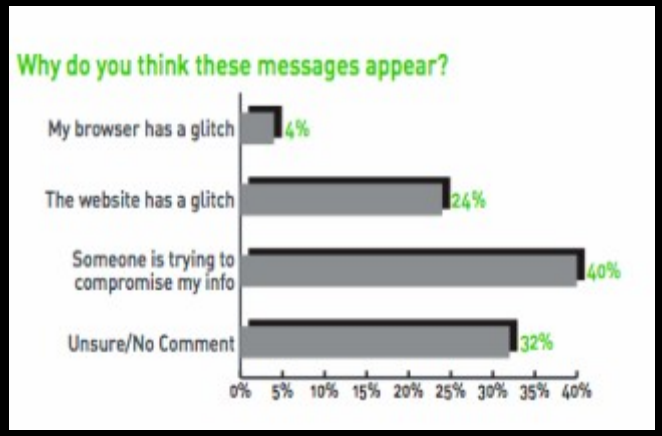
- First, you have to actually *use* it
  - How many executables will *you* download insecurely this week?
    - Attackers can pick and choose what domains to snipe, remember?
  - Who here still uses FTP? I do.
- Second, you must never downgrade.
  - http://www.foo.com 302'ing to https://www.foo.com – an attacker injected in the middle via DNS can replace the 302 with a 200. Will you notice?
    - Will your users?

IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES

More SSL

- Can't ignore the errors!
  - Data on top right is self-reported
    - Source: Consumer Reports
  - Actual data: When a major online bank in New Zealand had its cert expire, 99.5% of users still entered their credentials.
    - DNS-based attacker has a remarkably high success rate





Must Actually Care About Certificate Chain

- 327,467 SSL certificates were scanned
  - 140,355 SSL certificates were *self-signed* – 42%!
  - That's not even saying the other 58% are signed by a trusted CA!
- The *browsers* care about certificates
  - They worry about SSL meaning something
  - People who write applications – well, they might…

Who Says Applications Always (ever) Care About Cert Chains?

- Mike Zusman on SSL VPN's:
  - Yes, you will see many SSL VPN servers on the Internet serving invalid certificates.

    Cert validation varies product to product. One particular SSL VPN client I've worked with is hardcoded to only accept valid, trusted certs. If it is not signed by a trusted CA, the cert needs to be added to the local trust store. Another one allows you to turn validation on and off.

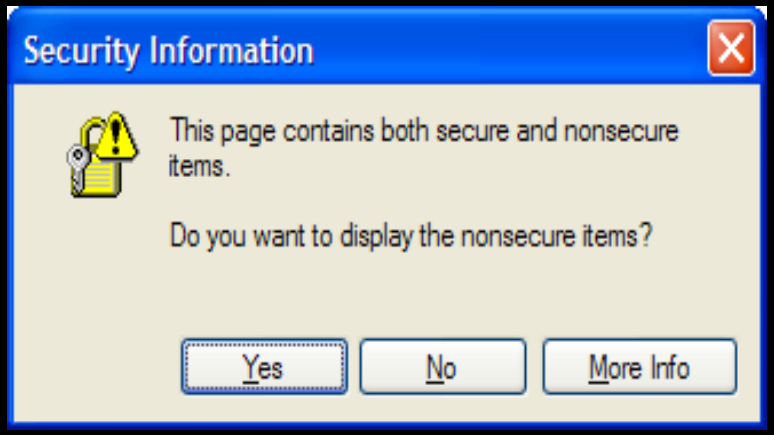
    None of the clients I've seen do any caching/white listing (like an SSH client). This way once you have the SSL VPN client installed, you can connect to ANY server you need to seamlessly. Great for re-purposing attacks.
- In other news, go read Mike Zusman's notes on SSL VPN's. He is doing some very important work.

Even if actually a web app, must handle secure cookies correctly

- Secure cookies: For SSL only
- But, check it out – anyone can write a secure cookie!
  - The specific use case that Adam Barth and I are working on is defining a new HTTP header that indicates cookie integrity. The Set-Cookie header provides confidentiality using the Secure flag, but the Cookie header provides no corresponding mechanism for indicating integrity; thus the server cannot distinguish cookies that were set over HTTP from those that were set over HTTPS.
    - Collin Jackson http://markmail.org/message/fvbw24nd7egokdlx
- So even *if* the SSL site secures its cookies, it's still potentially vulnerable to an attacker who partially overwrites with his own cookies
  - With DNS poisoning, it's now possible to inject cookies via the HTTP "version" of any HTTPS site

Must not mix Secure and Insecure

- About the only thing you can safely import is an invisible image
- Everything else ends up able to mess with you
  - Full DOM access: <script> loaders, such as ad scripts and trackers
  - Partial Page Access: <iframe> to foreign domain
- Don't worry, Firefox is making it harder to notice this every day
  - Average website – 10-20 domains – very hard to convert everything to SSL, people want to do this incrementally



IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES

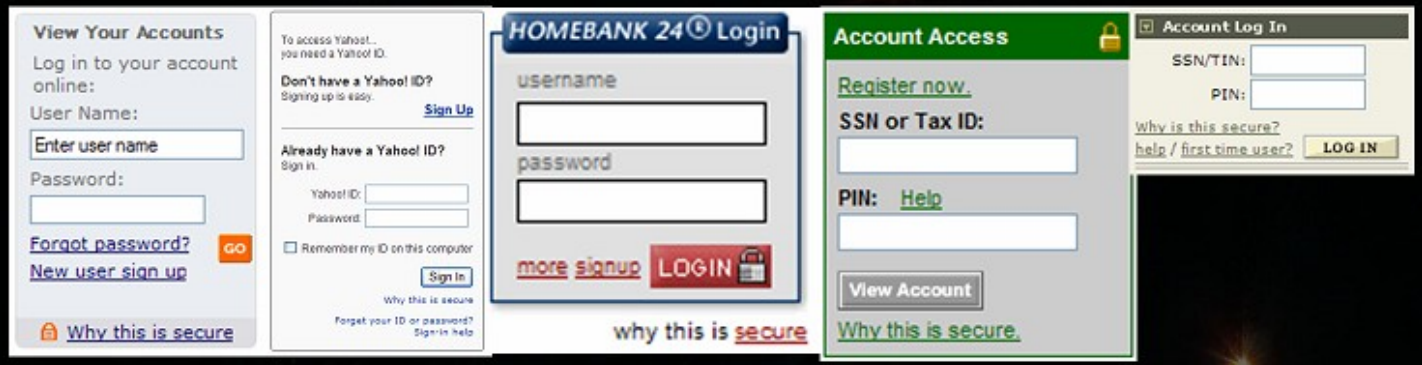
Woe To The Poor Flash Security Guy Who Had To Document AllowInsecureDomain()

- **http://tinyurl.com/6dv58r**
- Yes, the call is about as bad as it sounds
- The author of this document hates the call more than I do
- The author of this document was right – AllowInsecureDomain() + DNS hijack = useless SSL
- …and of course…

IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES

"Why Is This Secure"

The World's Most Depressing Google Search



- Everything here is delivered over HTTP. So an attacker can just replace https with http and hijack your login.
- 26% of the Top 50 banks operate insecurely; **all but one** use a picture of a lock to assure users the link is safe
  - 

We Live In The Future

- Gone from 26% in 2006 to ~5% maybe
  - Financial sites are improving, if not perfect
    - See: "Analyzing Web Sites For User-Visible Security Design Flaws", Atul Prakash, Laura Falk, Kevin Borders
    - (Things are a little better than they were back when this data was collected)
  - Everyone else still kinda sucks
- Still, lots of sites are trying the "we'll protect credentials by logging in via SSL, then switching back to HTTP game"
  - Graham and Maynor's Sidejacking works fine now
  - So does just stripping the SSL, or injecting Javascript to read the typed credentials from the DOM
  - Only now, you're not on the LAN – you're wherever
- Security is quantized, but people keep trying to make it incremental.

Cert should not use MD5

- Oh wait, it's 2008, we're still pretending this hashing algorithm works
  - Pay no attention to its 12 year old death row status
  - Or its government decertification a decade ago
  - Or the generation of collisions 3 years ago
  - Or the second preimage ("cloning") attacks against MD4 2 years ago
  - Etc. etc. etc.

Cert Must Never Have Been Generated By Debian

- People are comparing “Kaminsky DNS” to “Debian NRNG”
  - Since when was there a competition?
  - These bugs combine catastrophically!
    - 1) Almost all Debian-generated and CA signed certificates are still valid
      - Revocation is a myth, only expiration works
    - 2) Whoever collected all those public certificates when the bug broke, can *still* impersonate anyone with the cert
      - Up to five years!
    - 3) DNS flaw provides MiTM access such that Debian NRNG may be exploited
      - Why did everyone compute Debian NRNG OpenSSH keys, but not grab OpenSSL certs?
      - No obvious exploit path, until now

So?

- We've known all those things about SSL.
  - This is all old stuff.
  - Bla bla bla, show me something new, bla bla bla
- OK. Fine.
  - Why do you think SSL certificates are valuable?
    - Anyone can buy one
    - Anyone can generate bits
    - What prevents anyone in the audience from getting a cert for www.microsoft.com?

Into The Lions Den

- CA's sell bits
  - But there's some meaning applied to those bits – an assertion that an identity has been validated, at least to some level
  - How are identities validated by most CA's?

Say Hello To My Little Friend

- Domain Validation: How SSL Certificate Authorities use DNS to determine whether you get a certificate
  - Look up the domain in WHOIS
    - DNS address lookup
  - Send an email to the mail address on file
    - DNS MX record lookup
  - Visit the web page and look for a file
    - DNS A record lookup
- **Guess how secure that is in the face of a DNS attack?**

Hello My Little Friend

- Actually, we're doing OK
  - For some reason, every CA scrambled during the month of July to make sure that they were patched
    - Thank you, Tom Albertson, Kelvin Yiu, Zot O'Connor of Microsoft ☺
    - Thank you Verisign, Comodo, Digicert, Trustwave, everyone who kept this matter secret
- www.SSLShopper.com figured it out…but they think the answer is to buy EV certs. Unfortunately…

And what about EV?

- EV is a *display* mechanism, not a *code security* mechanism
  - Extended Validation. The browser's scripting policy does not distinguish between HTTPS connections that use an Extended Validation (EV) certificates from those that use non-EV certificates. For example, PayPal serves https://www.paypal.com/ using an EV certificate, but a principal who has a non-EV certificate for www.paypal.com can inject script into the PayPal login page without disrupting the browser's Extended Validation security indicators; see Figure 2.
    - "Beware of Finer Grained Origins", Collin Jackson, Adam Barth
- So, no. EV does nothing in the face of also-extant Domain-Validated Certificate

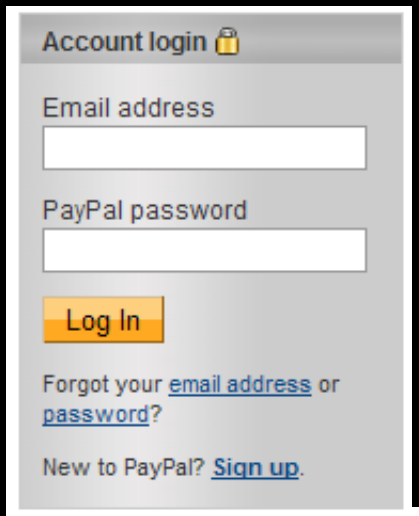
What Else Is Interesting?

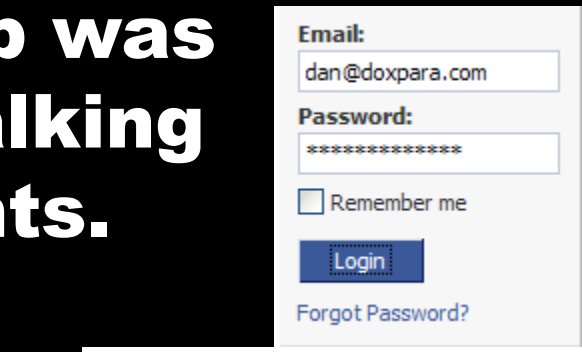
- CA's have web interfaces to manage previously issues certs…
  - …web interfaces you have to sign into.

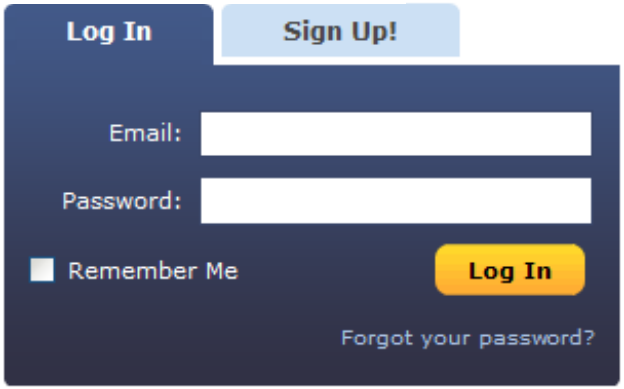
IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES

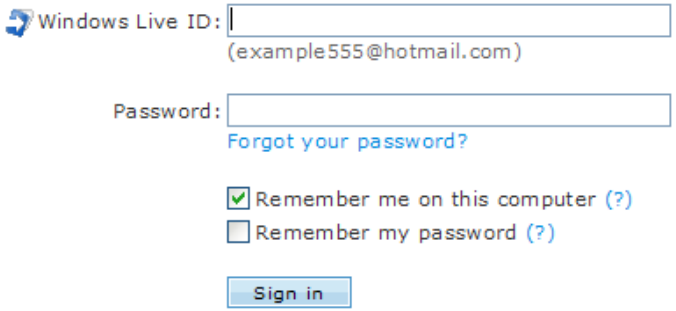
When I said The Web was broken, I wasn't talking about just its clients.

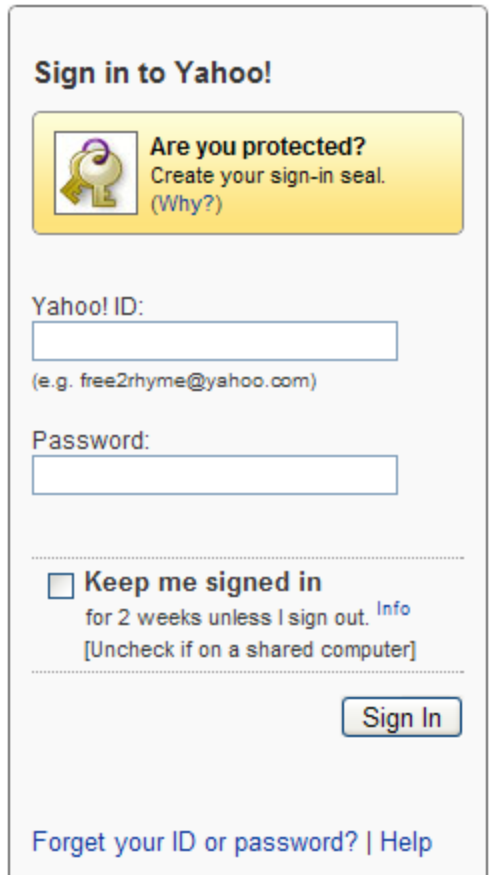
(confused?)

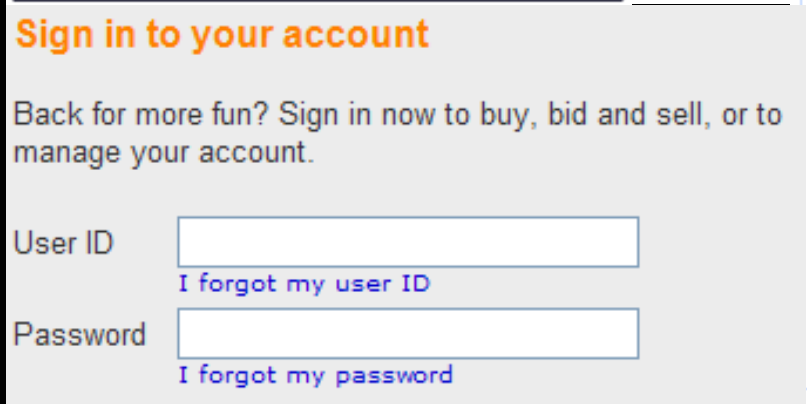


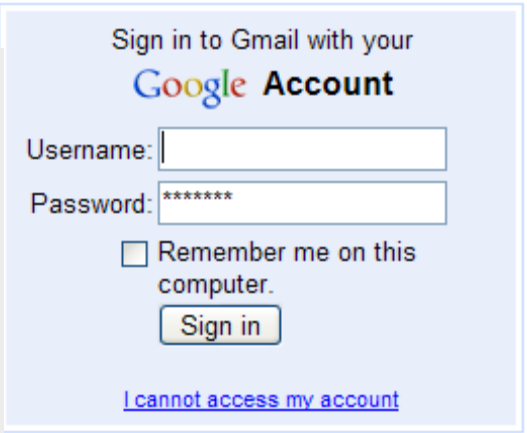










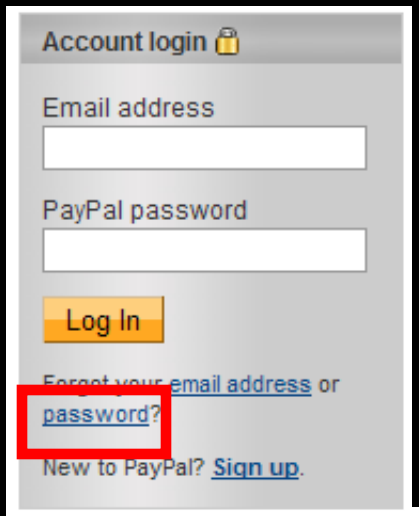


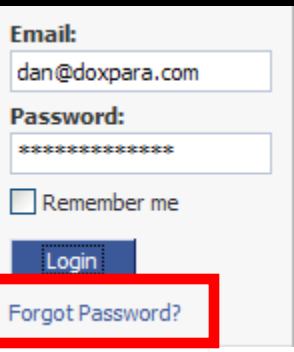
www.ioactive.com

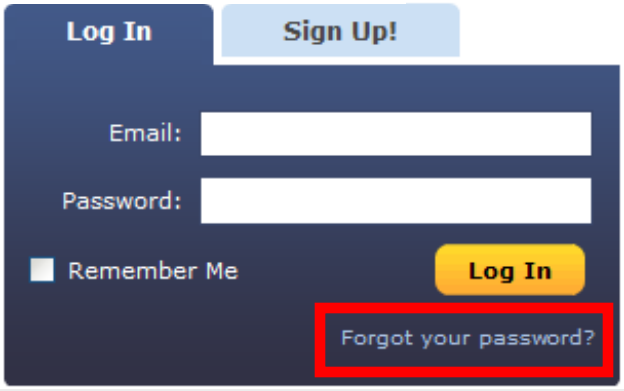
IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES

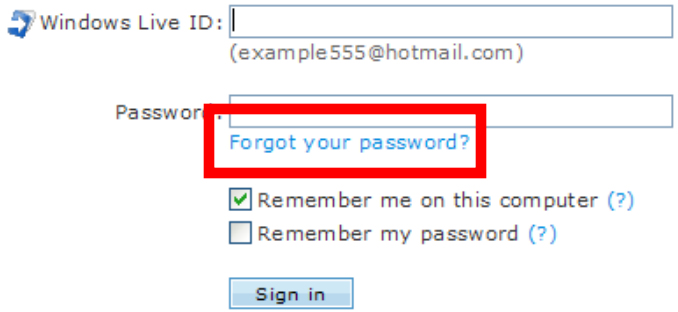
Welcome to the Skeleton Key.

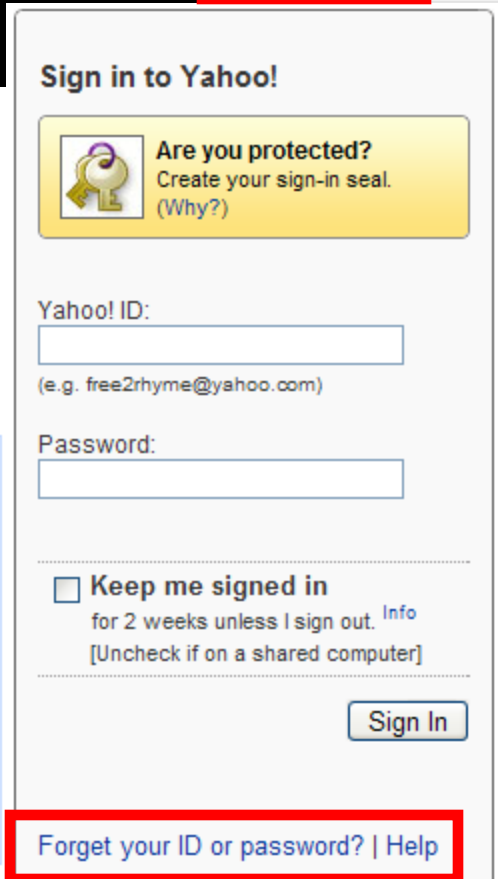
It's By Design.

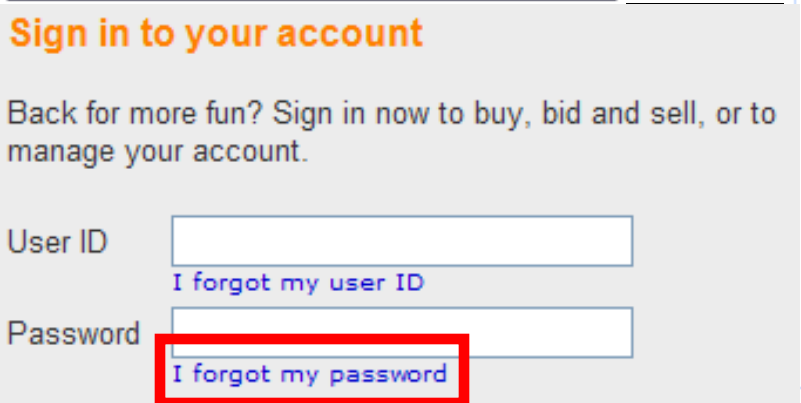


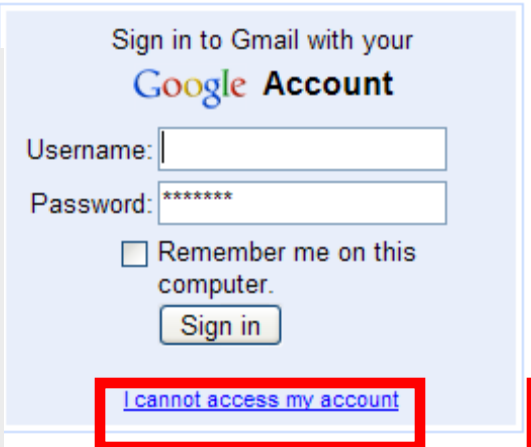












Forgot My Password Modes

- This is a generic lost credential technique
  - Generally, a fully automated way to get into an account without the password
    - Near-universally deployed
  - Three modes seen in the field
    - 1) Password Leak: Just mails you your password. Somewhat uncommon.
    - 2) Reset Password: Mails you a link that resets your password. Guarantees detection of attack. Most common.
    - 3) Reset w/ Additional Protections: Mails you a link, and makes you jump through hoops. Somewhat common on high-value sites.
  - #1 and #2 are trivial to pop (though #2 has side effects).

Attacking Forgot My Password systems

- It's just an email, meaning, it forces a lookup to an attacker controlled name
  - What did we need, to pollute com?
  - *This means any lookup can spawn any other arbitrary lookup, on demand*
    - *1. Force a lookup to 1.badguy.com*
    - *2. Reply with a referral (NS or CNAME) to 1.foo.com*
    - *3. Follow the reply with an immediate stream of fake replies from the foo.com name server*
- Not complicated. After poisoning, request password for arbitrary account
  - It will do an MX lookup
  - You will see the MX lookup
  - Game over ☹

News

- Fixed (beyond just the CA's)
  - Google
  - Live
  - Yahoo
  - Paypal
  - eBay
  - MySpace
  - Facebook
  - LinkedIn
  - Bebo
  - Craigslist
  - LiveJournal
  - Hi5
  - Citrix (GoToMyPC)

- This is very, very cool. Thank you to all the companies who worked with me on this!
  - Ow my cell phone bill ☺

IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES

Reality Check

- No way we got everyone

Results **1** - **10** of about **8,670,000** for "**forgot** my **password**". (**0.21** seconds)

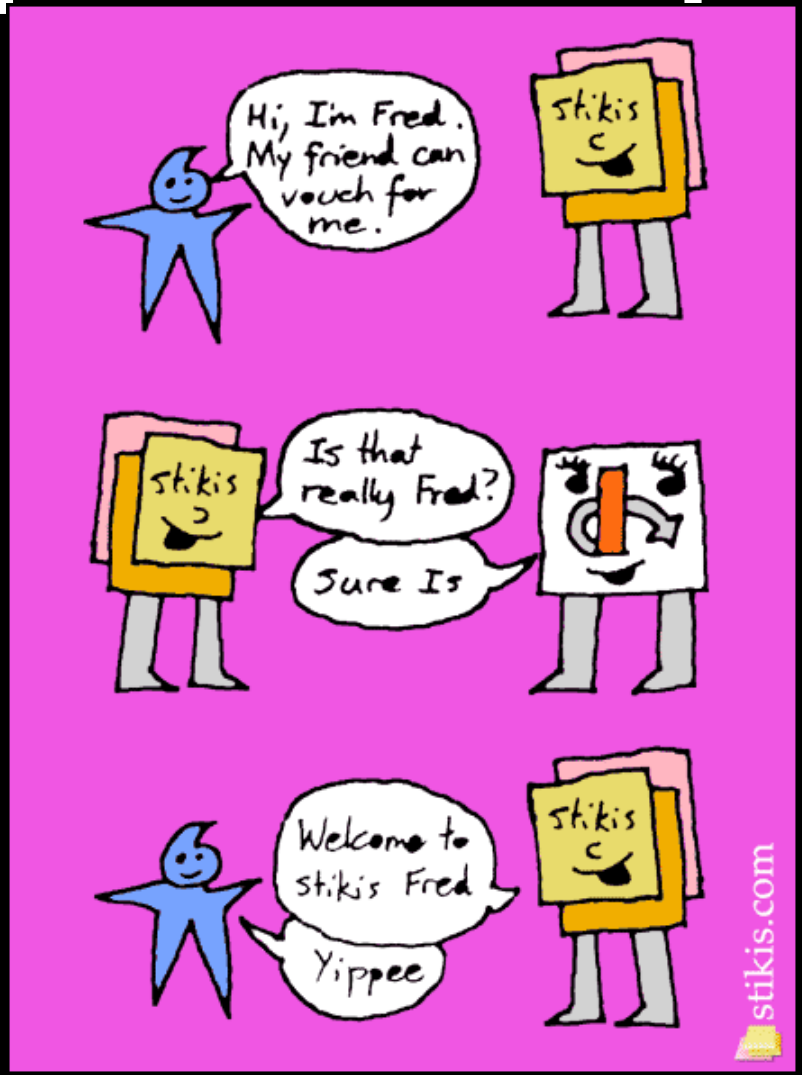
Results **1** - **10** of about **88,000,000** for "**forgot** your **password**". (**0.29** seconds)

Results **1** - **10** of about **69,600,000** for "**forgot password**". (**0.23** seconds)

- But we did ok.

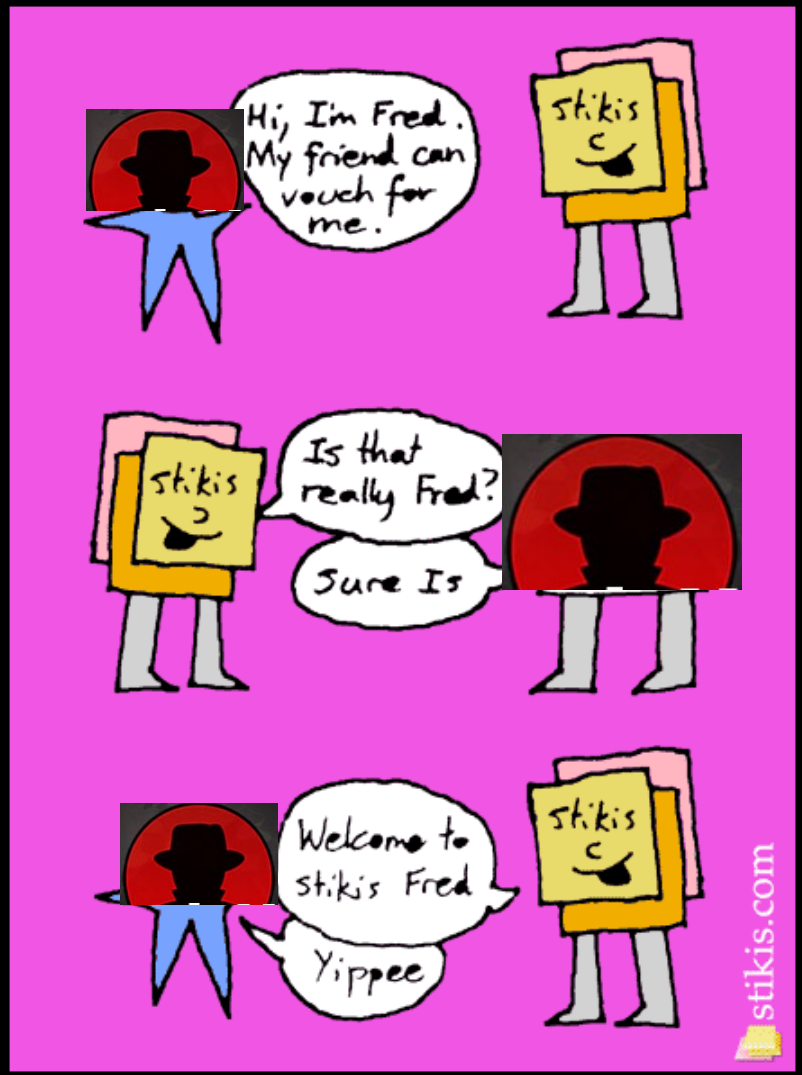
IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES

Would OpenID have helped?



IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES

How did Stikis find the “friend”?
Hint: DNS



What of OpenID with HTTPS?

- Should be fine…
  - Alas: Ben Laurie found that a couple major OpenID providers were using HTTPS…
    - But with a Debian misgenerated certificate
    - DNS + OpenID + Debian NRNG = WIN
- *glorious*

So Right About Now You're Probably Thinking...

- OK. Clients are hosed. Web 2.0 is hosed. But my intrawebs are safe!
  - Are they? Are they really?

Let Us Discuss The Inconvenient Matter Of Reverse DNS

- You know, we *also own in-addr.arpa*
  - This is the space that, when you look up 1.2.3.4, returns "a.b.c.d.com"
- What can you do with this?
  - Obvious: Spoof log entries in Apache
    - Apache Double-Reverse Lookup Log Entry Spoofing Vulnerability
      - Martin Kraemer, 2002
    - Apache will log the name that reverse DNS provides, *if* that name resolves back to the same IP
      - Well, we control both forward and reverse DNS, so heh
    - May even be able to fake numeric TLDs…
      - 6.6.6.6 IN PTR 1.2.3.4
      - 1.2.3.4 IN A 6.6.6.6
      - Possibly (probably) stopped by client side APIs
        - » Never assume an API is every smarter than it had to be to ship

More Reverse DNS

- SQL/Script Injection via Client-Targeted Reverse DNS
  - Reverse DNS is often put into a DB
  - Input may not be sanitized, because server may sanitize for the client
  - What if the client doesn't sanitize as much as the server?
    - This is probably pretty painful to execute

Lets Party Like It's 2007

- Black Ops 2007: Possible to use browser plugins to connect to internal resources
  - You browse to my site, I get TCP, maybe UDP, to your site
    - Flash
    - Java
  - Flash secured this with crossdomain.xml, per IP address
  - Java secured this with…reverse DNS
    - Which we own.
    - I can has 1.0.0.10.in-addr.arpa!
    - Full TCP and UDP from any browser in your org, to any host behind the firewall, if you don't patch DNS
      - And have Java
      - Bonus: IPsec!
    - Note: You don't get 127.0.0.1, because 1.0.0.127.in-addr.arpa has an authoritative record on most servers.
      - MAY get per-interface bind though…
    - To get 127.0.0.1 out of Java, see John Heasman's talk

Spreading The Phun

- If you have arbitrary socket access…
- …then you can spew packets on Port 53…
- …then you can scan for more name servers behind the corporate firewall.

✂☹

Also…SNMPv3

- If you can spoof arbitrary UDP packets, behind the firewall, you can spoof arbitrary SNMPv3 packets too
  - Have *you* patched against the SNMPv3 bug?

Enough with the client bugs?

- What about servers? Are they vulnerable?

Which would you rather own? BGP? Or DNS?

- BGP controls the flow of packets already destined for the Internet
  - Two boxes on the same LAN can not be attacked with BGP – packets won't even route
  - No specificity – can't select sessions opened at time A, but not time B
  - Lots of visibility – people see, and log, BGP announces
- DNS controls where packets are destined for in the first place
  - Two boxes on the same LAN will trust DNS to determine if they should talk directly to eachother – packets will be *made to route*
  - Some specificity – can select traffic on a per name, per query basis
  - Very low visibility – distributed, minimal logs

So DNS > BGP, right?

- DNS > BGP
  - Or is it?
  - Who says attacks compete?
  - Use BGP to hijack DNS traffic!
  - Stealing The Internet – No Really – 4PM, Track 4, Kapela/Pilosov
    - Only works if you have BGP connectivity, but do you trust everyone with BGP connectivity?
      - Pakistan 1, YouTube 0

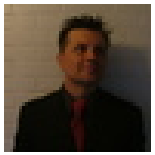
Difficulty: Cannot poison authoritative on servers…

- …with DNSRake, anyway.
- Can still poison authoritative with client attacks
- Not everyone is talking directly to primaries/secondaries
  - Isolated Split Brain is *very* safe
    - No external DNS to desktop + fake roots = attacker never sees any internal traffic
  - AD is heavy on primaries – also safe
    - No cache to corrupt
  - Split brain environments are heavy on forwarders – somewhat safe
    - Attacker may not be able to spoof destination of forwarder
  - If internal servers are talking to a normal name server, that's recursing both to the Internet and to internal names: *Game On*

IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES

When Internal DNS Goes Bad

- *So much bad behavior behind the firewall, all directed via DNS!*
  - Telnet
  - SNMP – queries and traps
  - Auth servers (RADIUS/TACACS)
  - Backup/Restore
  - SOA architectures
    - Resolve back to *names*, DNS determines *addresses*
  - Backend Databases



shawnmoyer: @hevnsnt I think something about DSNs. Apparently @dakami found a way to hijack everyone's **ODBC** connections, or something. Plz advise.

about 16 hours ago · Reply · View Tweet

Even if *internal* DNS is hard to hit, external dependencies are fair game

- *So many connections* between companies
  - DNS controls how the servers find each other
  - The link *might* be secured
    - If SSL is used – is anyone actually checking the certificates?
      - Are you sure?
    - If IPsec is used – is it tied to destination subnet?
      - **DNS changes destination subnet, therefore DNS can change IPsec rules.**

The ultimate external dependencies

- Payment processing / Offsite backup
  - *Now is not a good time to have an insecure link to your offsite stores*
  - I'm not saying anyone does. But if there's a scintilla of a chance, *patch patch patch*
- SNMP against the Internet
  - If you are using SNMP to log into machines on the Internet, you're probably using DNS to find them
    - Interesting interactions with SNMPv3 bugs?
- Search Engine Population
  - There's Search Engine Optimization, and there's this
- CDN.

Content Distribution Network Corruption

- How do you think you populate a CDN?
  - Provide the CDN a URL
    - Uses DNS, pulls data
  - Treat the CDN as a proxy
    - Uses DNS, pulls data
  - *It would be really, really bad if Akamai etc. DNS went bad*

Summary

- DNS servers had a core bug, that allows arbitrary cache poisoning
  - The bug works even when the host is behind a firewall
  - There are enough variants of the bug that we needed a stopgap before working on something more complete
- Industry rallied pretty ridiculously to do something about this, with hundreds of milllions protected
- DNS clients are at risk, in certain circumstances
- We are entering (or, perhaps, holding back a little longer) a third age of security research, where all networked apps are “fair game”
  - Autoupdate in particular is a mess, broken by design (except for Microsoft)
- SSL is not the panacea it would seem to be
  - In fact, SSL certs are themselves dependent on DNS
- DNS bugs ended up creating something of a “skeleton key” across almost all major websites, despite independent implementations
- Internal networks are not at all safe, both from the effects of Java, and from the fact that internal routing could be influenced by external activity
  - The whole concept of the fully internal network may be broken – there are just so many business relationships – and, between IPsec not triggering and SSL not being cert-validated, these relationships may not be secure
  - We’re not even populating CDN’s securely!

Hype

- So I hear there are people who don't think breaking DNS is a big deal ☺
  - I hear them pretty loudly.
    - So did the Pwnie judges – apparently the overwhelming majority of Most Overhyped noms were for DNS bugs (mine and Petka's)
  - I do believe they called me out.
  - I humbly nominate them for Most Overhyped Bug.
    - They said there was a bug in my talk – that it was old.
    - Halvar did not call me out. He is very much not nominated.

Lessons Learned

- We have to get better at fixing infrastructure.
  - We got lucky with this bug.
  - The next one will not be so “smooth”
  - Disaster recovery planning needs to include how to handle the discovery in a flaw *in any mission critical code anywhere*
    - Servicability needs to start becoming a more important purchasing metric.
    - Servicability is, ultimately, the measure of software flaw survivability.
- Cooperation across competitors, and researchers, can indeed be very productive
  - 120M users from just one infrastructure provider
- A lot of people just do not realize the degree to which security best practices have been ignored for years
  - DNS should not have been capable of this much damage.
  - It was. Why?

Bottom Line

- We are doing a lot of things insecurely.
  - Even with DNS fixed, there are other scenarios in which unencrypted IP traffic is lost to an attacker
  - That attacker is capable of way more than he should be.
    - More than I've even said here.